B2

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-259643

(43)Date of publication of application : 13.09.2002

(51)Int.Cl. G06F 17/60
G06F 9/44

(21)Application number : 2001-057621

(22)Date of filing : 02.03.2001

(71)Applicant : HITACHI LTD

(72)Inventor : MURASE ATSUSHI
DOI KOJI
TANAKA TETSUO

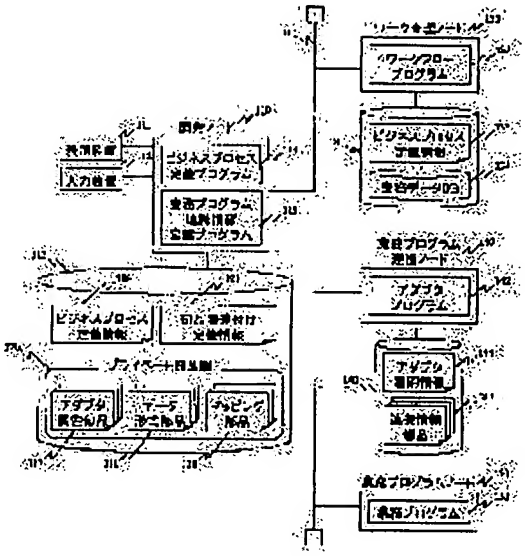
(54) BUSINESS PROCESS CONTROL PROGRAM

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a system development environment equipped with a user interface capable of diverting application link definition information for a fine unit and clearly grasping application link information in a program for executing different kinds of application for processing each of jobs with the control of business process.

SOLUTION: The application link information is managed as a component while being classified to three of executing system 117, form 118 of data to be dispatched, correspondent relation and conversion rule 119 of data items. The generation of definition information 121 relating a work node on the business process and this component is a system, with which a work program link information definition program 115 makes a user select the icon of the component. Besides, the component information is stored in a component cabinet and is referred to from a plurality of development nodes 110. Further, when the user performs unjust definition operation or the like, the work program link information definition program 115 raises an alarm to the user.

【図1】



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2002－259643
（P2002－259643A）
(43)公開日　平成14年９月13日(2002.9.13)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード（参考）
Ｇ０６Ｆ　17/60	１６２	Ｇ０６Ｆ　17/60	１６２Ｃ	５Ｂ０７６
9/44		9/06	６２０Ａ	

審査請求　未請求　請求項の数４　ＯＬ　（全 18 頁）

(21)出願番号　特願2001－57621(P2001－57621)

(22)出願日　平成13年３月２日(2001.3.2)

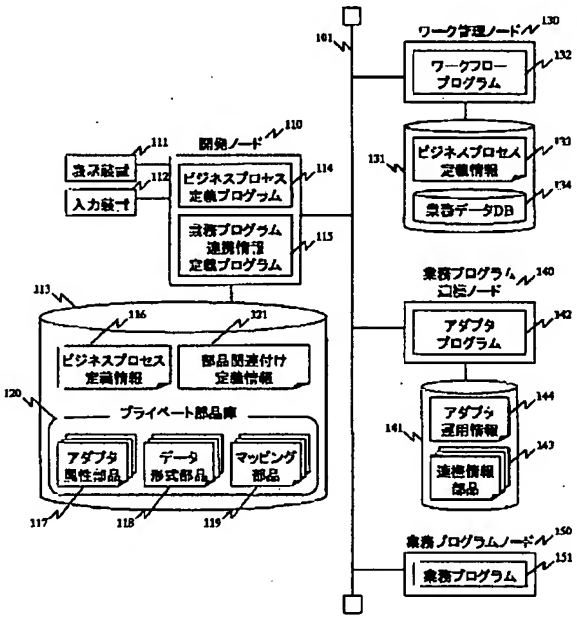
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　村瀬　敦史
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　土井　宏治
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(74)代理人　100075096
弁理士　作田　康夫

最終頁に続く

(54)【発明の名称】　ビジネスプロセス制御プログラム

(57)【要約】　（修正有）
【課題】ビジネスプロセスの制御に伴い個々の業務を処理する異種アプリケーションを実行するプログラムにおいて、アプリケーション連携定義情報を細かな単位で流用できること、アプリケーション連携情報を明確に把握できるユーザインタフェースを備えたシステム開発環境を提供する。
【解決手段】アプリケーション連携情報は、実行方式１１７、受け渡すデータの形式１１８、データ項目の対応関係と変換ルール１１９の三つに分類し部品として管理する。ビジネスプロセス上の業務ノードと該部品の関連付け定義情報１２１の生成は、業務プログラム連携情報定義プログラム１１５が、ユーザに部品のアイコンを選択させる方式とする。また、部品庫に該部品情報を格納し、これを複数の開発ノード１１０から参照する。また、業務プログラム連携情報定義プログラム１１５は、ユーザが不当な定義操作をした場合などにユーザに警告を発する。

【図1】



【特許請求の範囲】
【請求項１】複数の業務から構成されるビジネスプロセスを実行するための情報処理をコンピュータに実行させるための制御するコンピュータプログラムであって、
前記複数の業務のそれぞれを実行するための各業務プログラムを識別する識別情報および前記業務プログラムを実行するための実行条件を定義するアダプタ機能と、
前記複数の業務プログラムのうち、前記ビジネスプロセスを実行するために互いに連携する業務プログラム間で送受信するデータの形式であるデータ形式を定義するデータ形式定義機能と、
送受信される前記データの、連携する前記業務プログラムにおける第１の業務プログラムと第２の業務プログラム間での前記データ項目の対応関係および変換方式を定義するマッピング機能と、
定義された前記実行条件、前記データ形式、前記対応関係および前記変換方式を前記コンピュータに格納する機能と、
格納された前記実行条件、前記データ形式、前記対応関係および前記変換方式に基づいて、連携する前記業務プログラムにおける第３の業務プログラムおよび第４の業務プログラムの連携する機能を実現するためのビジネスプロセス制御プログラム。
【請求項２】請求項１に記載のビジネスプロセス制御プログラムにおいて、さらに、
格納された前記実行条件、前記データ形式、前記対応関係および前記変換方式を、変換することにより、各業務プログラム連携するための連携部品情報を作成する機能を有し、
前記連携する機能は、前記連携情報部品を用いて、前記連携する機能を実行するビジネスプロセス制御プログラム。
【請求項３】請求項２に記載のビジネスプロセス制御プログラムにおいて、さらに前記連携部品情報に対応する表示を、前記コンピュータが備えた表示装置に表示する機能を有するビジネスプロセス制御プログラム。
【請求項４】請求項３に記載のビジネスプロセス制御プログラムにおいて、
前記表示する機能は、前記表示装置に前記連携部品情報に対応するアイコンを表示するビジネスプロセス制御プログラム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、ワークフローシステムに接続されたアプリケーションをフローに伴い順次実行する技術、実行する異種アプリケーション間でデータの受け渡しを行う技術、該処理を実行するための定義情報生成を支援する技術に関する。
【０００２】
【従来の技術】ビジネス形態の変化が加速する中、企業活動を支えるコンピュータシステムの変化への対応速度が問われている。これまで企業内システムを再構築する場合は、多くの修正と新規アプリケーションの導入に伴う労力が必要であった。これを解決するため、ＥＡＩ（エンタープライズ・アプリケーション・インテグレーション）技術が注目されている。ＥＡＩ技術は、異種アプリケーションを連携するための技術であり、既存ソフトウェア資産の流用や有益な異種流通パッケージソフトウェアの組合せを実現することにより、企業内システムの早期構築を可能にする。 特開２０００－２０７４７４号公報（特許第３００６６１０号）には、ビジネスプロセスをワークフローエンジンで制御するプログラムにおいて個々の業務を処理する業務プログラムを連携するための技術が開示されている。本従来技術では、ビジネスプロセスを、アイコンを用いて簡易に定義するユーザインタフェースと、ビジネスプロセス上の業務ノードに対応する業務プログラムの受け渡しデータ、実行端末機アドレス、アクセス権限等の属性情報をノード単位に定義する手段と、ワークフローエンジンと業務プログラム間のデータ授受を実現するアプリケーションコントロール手段を備えることで、ワークフローシステムの構築・再構築に要するコストと手間を低減することを可能にしている。また、上記属性情報を部品管理することで他のワークフロー構築での再利用を実現している。
【０００３】
【発明が解決しようとする課題】しかしながら、上記発明では、一つの業務ノードに連携する業務プログラム属性情報の全てを纏めて管理するために、次に挙げる問題が生じる。
（１）データ形式のみ、データ項目の対応関係のみといった細かな定義内容の流用ができない。
（２）既存システムで定義したデータ形式のみを取込み流用するといった拡張に不向きである。
（３）同一の業務プログラムであっても、異なるデータ入出力を行う場合は、その都度上記アプリケーションコントロールの開発が必要となる。
【０００４】また、他の従来技術として、ビジネスプロセスに業務プログラムを連携するための支援システムが存在するが、ビジネスプロセスの定義手段と業務ノードに関連付ける業務プログラムの連携情報定義手段が分離しており、その対応関係を明示的にユーザに示すユーザインタフェースを提供するシステムは存在しないため、業務ノードと業務プログラム連携情報との関連性をユーザが把握しづらいという問題が発生している。
【０００５】そこで本発明では、上記課題を鑑みて、より細かな業務プログラム連携情報の流用を可能にすることで、更なるシステム構築期間の短縮が可能であり、ユーザがビジネスプロセス上の業務ノードと関連する業務プログラム連携情報との関係を明確に把握できるユーザインタフェースを備えた異種システム連携制御プログラ

ムの提供を目的とする。

【0006】

【課題を解決するための手段】本発明では、データ形式、プログラム属性およびこれらの対応関係・変換ルールを部品化し、部品化されたものとワークフローで実現するビジネスのプロセスの構成要素を関連付けるものである。

【0007】より、詳細には、（1）業務プログラム連携情報を業務プログラムの実行方式、受け渡すデータの形式、データ項目の対応関係と変換ルール、の三つに分類し部品として管理する手段と、（2）ビジネスプロセス上の業務ノードと（1）の連携情報部品を関連付ける手段と、（3）ビジネスプロセス定義をビジュアルに行う手段と連動した画面上にてユーザが部品を選択することで（2）の関連付けを行うビジュアルな連携情報定義手段と、（4）（1）の連携情報部品をビジネスプロセスとは独立な部品庫に格納・取り出しすることで、同一のビジネスプロセス内、あるいは、異なる複数のビジネスプロセス間において上記部品を流用することが可能な手段を備えることで、上記課題を解決する。

【0008】

【発明の実施の形態】以下に、本発明の実施形態について図面を用いて説明する。まず、本発明の第1の実施例（実施例1）を説明する。図1に本発明に関するシステム全体の構成例を示す。本システムは、ネットワーク101に接続された、ビジネスプロセスと業務プログラム連携情報を定義・格納する開発環境ノード110と、定義したビジネスプロセスを実行するワーク管理ノード130と、業務プログラムの実行と業務プログラムが受け渡すデータの変換を行う業務プログラム連携ノード140と、個々の業務を処理する業務プログラムノード150から構成される。

【0009】開発ノード110は、ビジネスプロセスや業務プログラム連携情報の定義画面を表示するための表示装置111と、定義作業の指示を入力するための入力装置112と、定義情報を格納するディスク装置113と、ビジネスプロセスを視覚的ユーザインタフェースにて定義するビジネスプロセス定義プログラム114と、業務プログラム連携情報を視覚的ユーザインタフェースにて上記ビジネスプロセス定義プログラム114と連動しながら定義する業務プログラム連携情報定義プログラム115から構成される。ディスク装置113には、ビジネスプロセスの業務ノードや遷移を定義したビジネスプロセス定義情報116と、業務プログラムを実行するアダプタプログラム142の属性を定義したアダプタ属性部品117と、業務プログラム間で受け渡すデータの形式を定義したデータ形式部品118と、受け渡すデータの項目間の対応関係と変換ルールを定義したマッピング部品119と、上記3種類の部品を格納するプライベート部品庫120と、上記部品とビジネスプロセスの業務ノードの関連付けを定義した部品関連付け情報121を格納する。

【0010】業務プログラムノード150は、個々の業務を処理する業務プログラム151から構成される。

【0011】業務プログラム連携ノード140は、業務プログラム151を実行するために必要な運用情報を格納するディスク装置141と、業務プログラム151を実行するアダプタプログラム142から構成され、ディスク装置141には、上記開発ノード110にて定義したアダプタ属性部品117、データ形式部品118、あるいは、マッピング部品119である連携情報部品143と、業務プログラムの入出力データを変換するためにアダプタプログラム142が参照するアダプタ運用情報144を格納する。

【0012】ワーク管理ノード130は、ビジネスプロセスを実行するために必要な情報を格納するディスク装置131と、ビジネスプロセスの実行を制御するワークフロープログラム132から構成され、ディスク装置131には、開発ノード110にて定義したビジネスプロセス定義情報133と、ビジネスプロセスを実行中に扱う業務データを管理する業務データDB134を格納する。

【0013】図2に、ビジネスプロセス定義プログラム114にて表示するビジネスプロセス定義画面200と生成するビジネスプロセス定義情報116の構造を示す。

【0014】ビジネスプロセス定義プログラム114は、画面上に業務ノードや遷移を示す矢印を配置することでユーザが視覚的にビジネスプロセスを定義できるユーザインタフェースを提供する。ビジネスプロセスの開始を示すソースノード201と終了を示すシンクノード202の間に、各業務を示す業務ノード203、204とそれらの遷移を示す矢印205、206、207を、マウスを用いて接続することでビジネスプロセスを定義する。また、ビジネスプロセス実行中に扱う業務データ208の形式もマウス操作にて定義する。

【0015】定義されたビジネスプロセス定義情報116は、業務ノードテーブル210、遷移テーブル220、業務データテーブル230として格納する。

【0016】業務ノードテーブル210は、各ノード属性を格納するテーブルであり、画面上に配置した各ノードの名称211と、ノートの種別212と、そのノードに連携させる業務プログラム151を実行するアダプタプログラムの名称142から構成される。例えば、本テーブルの2行目は画面上の業務ノード203の定義内容であり、本ノードが「見積」という名称で、業務を実行する作業ノードであり、「Adp001」という名称のアダプタプログラムによって目的の業務プログラムを実行することを示す。

【0017】遷移テーブル220は、画面上のノード間

を接続している遷移（矢印）の属性を格納するテーブルであり、遷移の名称２２１と、遷移元となるノードの名称２２２と、遷移先となるノードの名称２２３から構成される。例えば、本テーブルの１行目は画面上の遷移２０５の定義内容である。定義内容は、本遷移が「Ａｒｃ０１」という名称であり、ソースノード２０１から遷移が開始され、業務ノード（見積）２０３で終了する遷移することを示す。

【００１８】業務データテーブル２３０は、業務データの形式を格納するテーブルであり、業務データの名称２３１と、データの型２３２と、データのサイズ２３３から構成される。例えば、本テーブルの３行目は業務データ「数量」が整数型で４バイトサイズであることを示す。図２で示したように、ユーザは画面上にアイコンや矢印を配置する操作と名称等の入力のみによってビジネスプロセスの定義を行うことができ、定義したビジネスプロセス定義情報１１６が開発環境ノード１１０のディスク１１３に格納される。

【００１９】図３に、ビジネスプロセスの業務ノードと業務内容を処理する業務プログラムの関連付けの構造、及び、生成する部品関連付け情報１２１の構造を示す。

【００２０】本実施例では、業務プログラムを連携するための情報を３種類に分類し部品化する。連携する業務プログラムを実行するアダプタプログラムの実行方式や入出力データといった属性を持つアダプタ属性部品３０１、３０２と、業務プログラムが入出力するデータやビジネスプロセスが保持する業務データのデータ形式を持つデータ形式部品３０３、３０４、３０５と、ビジネスプロセス上の業務データと業務プログラムが入出力するデータの間で受け渡すデータ項目の対応関係と変換ルールを持つマッピング部品３０６、３０７の３種類である。

【００２１】ビジネスプロセスと業務プログラムの連携定義は、上記部品とビジネスプロセスの業務ノードとの関連付け、上記部品と業務プログラムを実行するアダプタプログラムとの関連付け、という２段階の関連付けで行う。例えば、アダプタ属性部品３０１と業務プログラム３１１を実行するアダプタプログラム３１２を関連付ける。次に、業務ノード３２１と該アダプタ属性部品３０１を関連付けることにより、業務ノード３２１に対する業務プログラム３１１の連携定義を行う。

【００２２】業務プログラムとビジネスプロセス間で受け渡すデータに関しては、業務プログラムの入出力データの形式と、業務データの形式と、それらの間のデータ項目対応関係と変換ルールを定義する。例えば、業務プログラム３１１が出力するデータの形式をデータ形式部品３０３に定義し、更に該データ形式部品３０３を、業務プログラム３１１を実行するアダプタプログラム３１２に関連付けたアダプタ属性部品３０１に関連付ける。一方、ビジネスプロセスの業務データ３２２の形式をデータ形式部品３０４に定義し、業務データ３２２に関連付ける。そして、上記で定義した業務プログラム３１１の出力データ形式３０３を変換元とし、業務データ形式３０４を変換先とする、マッピング部品に該二つのデータ形式部品３０３、３０４を関連付け、各データ項目の対応付けとその際必要となる変換ルールを定義する。このようにして、業務プログラム３１１が出力するデータを変換し業務データ３２２に格納する関係を定義する。

【００２３】定義した部品関連付け情報１２１は、業務データ部品テーブル３３０、業務ノード連携属性テーブル３４０として格納する。

【００２４】業務データ部品テーブル３３０は、業務データに対応するデータ形式部品のＩＤを格納するテーブルである。例えば、図３の例ではデータ形式部品３０４が業務データの形式として関連付けられていることを示す。業務ノード連携属性テーブル３４０は、業務ノードに関連付けられたアダプタ属性部品とマッピング部品のＩＤを格納するテーブルであり、対象となる業務ノードの名称３４１と、該業務ノードに関連付けられたアダプタ属性部品のＩＤ３４２と、該業務ノードに連携する業務プログラムの入力データと業務データの変換を示すマッピング部品のＩＤ３４３と、同じく出力データの変換を示すマッピング部品のＩＤ３４４から構成される。例えば、本テーブルの１行目は業務ノード（見積）３２１にＩＤ「Ａ１」のアダプタ属性部品３０１と、ＩＤ「Ｍ１」のマッピング部品３０６が出力データに関して関連付けられており、入力データに関してはマッピングが特にないことを示す。

【００２５】アダプタ属性部品に関連付けたアダプタプログラム、アダプタ属性部品の入出力データ形式、マッピング部品の変換元（先）データ形式等の部品間の関連付け情報は各部品の属性として保持する。

【００２６】図３で示したように、本発明では業務プログラム連携情報を部品化し、該部品をビジネスプロセス上の業務ノード、更に、業務プログラムを実行するアダプタプログラムと関連付けることによって、ビジネスプロセスへの業務プログラム連携情報を定義し、定義した部品関連付け情報１２１を開発環境ノード１１０のディスク１１３に格納する。

【００２７】図４に、ビジネスプロセスの業務ノードに上記アダプタ属性部品やマッピング部品を関連付けるための画面を示し、業務ノードへの連携情報部品の関連付け手順を説明する。

【００２８】ここでは、ビジネスプロセス定義画面２００に表示されている業務ノード（見積）２０３にアダプタ属性部品を関連付ける場合について説明する。まず、ビジネスプロセス定義画面２００上の業務ノードを示すアイコンをマウスで選択し、属性設定の中から「アダプタ属性」を選択することで、アダプタ属性部品選択画面４００が表示される。該画面には既に定義されたアダプ

タ属性部品の一覧（４０１、４０２、４０３）がアイコンとして表示される。次に、ユーザは、ビジネスプロセス定義画面２００で選択した業務ノード２０３に関連付けるアダプタ属性部品を一覧の中より選択する。図４の例では、「Ａ２」というアダプタ属性部品を選択した状態を示している。そして、選択ボタン４１１をクリックすることでアダプタ属性部品選択画面の表示が終了し、業務ノード２０３とアダプタ属性部品４０２の関連付け定義が完了する。この際、上記業務ノード連携属性テーブル３４０に関連付け情報が格納される。データ形式部品やマッピング部品の関連付けも同様な手順で行う。

【００２９】このように、本発明では、ビジネスプロセスの構成要素に対する業務プログラム連携情報を、画面上に表示された部品アイコンの選択という視覚的な操作によって定義する手段を提供する。このことにより、ユーザはその関係を明確に把握することが可能である。

【００３０】図５にビジネスプロセスの構成要素に連携情報部品を関連付ける際の処理フローを示す。まず、ビジネスプロセス定義プログラム１１４によってビジネスプロセス定義画面２００が表示されている（５０１）。ユーザが画面上の業務ノードを選択し目的の属性設定を選択することで業務プログラム連携情報定義プログラム１１５が起動される。

【００３１】次に、業務プログラム連携情報定義プログラム１１５が目的の連携情報部品を選択する処理フローについて説明する。５０２では、ユーザがビジネスプロセス定義画面にて選択した業務ノードに指定の部品を関連付ける事が可能か否かを判定する。例えば、ユーザがソースノード２０１を選択した場合は、これに連携情報部品を関連付けることができないため、警告メッセージを表示して（５０３）業務プログラム連携情報定義プログラムの実行を終了する。本発明では、このように処理５０２、５０３によってユーザの不当な部品関連付け操作を自動的に回避する手段を提供している。

【００３２】ユーザが選択した業務ノードが部品を関連付ける対象として妥当な場合、処理５０４に遷移する。５０４では、関連付ける部品の種別を判定する。ユーザが属性設定として選択した項目によって処理を分岐する。ユーザが「アダプタ属性」を選択した場合は処理５０５に遷移し、アダプタ属性部品選択画面を表示してユーザに所望の部品を選択させる。同様にして、「データ形式」を選択した場合は、処理５０６に遷移してデータ形式部品選択画面を、「マッピング」を選択した場合は、処理５０７に遷移してマッピング部品選択画面を表示して所望の部品をユーザに選択させる。処理５０５、５０６、５０７によってユーザが所望の部品を選択すると、それぞれの部品選択画面の表示を終了し、選択した部品のＩＤを業務ノード連携属性テーブル、あるいは、業務データ部品テーブルに格納する（５０８）。このような処理によって業務ノードへの連携情報部品の関連付けを行い、業務プログラム連携情報定義プログラムの実行を終了する。

【００３３】図６に、連携情報部品が保持する属性を示す。まず、連携情報部品一覧テーブル６００にて定義されている全ての部品の一覧情報を管理する。連携情報部品一覧テーブル６００は、定義した部品の種別６０１と、該部品のＩＤ６０２と、該部品の詳細属性が格納されるテーブル名６０３から構成される。例えば、本テーブルの１行目は「Ａ１」というＩＤを持つアダプタ属性部品が存在し、「ＴＢＬＡ００１」というテーブルに詳細属性が格納されていることを示す。

【００３４】アダプタ属性部品属性テーブル６１０は、一つのアダプタ属性部品の属性を格納するテーブルであり、部品の名称６１１と、部品のＩＤ６１２と、対応するアダプタプログラムのＩＤ６１３と、対応する業務プログラムへの入力データ形式を示すデータ形式部品のＩＤ６１４と、同じく出力データ形式を示すデータ形式部品のＩＤ６１５から構成される。

【００３５】データ形式部品属性テーブル６２０は、一つのデータ形式部品の属性を格納するテーブルであり、部品の名称６２１と、部品のＩＤ６２２と、該データ形式部品に定義されたデータ項目の属性から構成される。一つのデータ形式部品には、複数のデータ項目の定義を格納できる。一つのデータ項目に関して、項目の名称６２３と、データの型６２４と、データのサイズ６２５を格納する。

【００３６】マッピング部品属性テーブル６３０は、一つのマッピング部品の属性を格納するテーブルであり、部品の名称６３１と、部品のＩＤ６３２と、変換元データの形式を示すデータ形式部品のＩＤ６３３と、変換先データの形式を示すデータ形式部品のＩＤ６３４と、データ項目の対応関係を示すテーブルの名称６３５から構成される。データ項目マッピングテーブル６３６は、該マッピング部品属性テーブル６３０の項目６３５に対応するテーブルであり、変換元となる項目の名称６３７と、変換先となる項目の名称６３８と、変換ルール６３９から構成される。

【００３７】図７にデータ形式部品を定義する画面を示し、その手順について説明する。データ形式部品選択画面７００は、図４で説明したように、ユーザが所望の部品を選択するための画面である。部品選択の際、ユーザの所望する部品が未定義であれば部品を新規生成する。生成した新規部品はデータ形式部品選択画面７００の部品一覧にアイコンが追加表示されるので、ユーザは該部品を選択することができる。

【００３８】新規部品の生成手順について説明する。まず、データ形式部品選択画面７００の追加ボタン７０１をクリックするとデータ形式部品属性定義画面７１０が表示される。次に、該画面上で必要なデータ形式部品の属性を入力する。最後に、該画面を閉じることでデータ

形式部品が新規生成され、データ形式部品選択画面７００に生成した部品のアイコンが追加表示される。

【００３９】定義した部品の属性を変更する場合は、データ形式部品選択画面７００の部品一覧から対象のアイコンを選択し編集ボタン７０２をクリックする。新規生成と同様にデータ形式部品属性定義画面７１０が表示されるので必要な属性項目を修正し、該画面を閉じることで部品属性の編集が完了する。

【００４０】データ形式部品選択画面７００の一覧から部品を選択し削除ボタン７０３をクリックすることで、指定部品を削除する。このように、指定した部品のアイコンが一覧から削除される。

【００４１】このように、本発明では各連携情報部品の生成、属性定義についてもプログラミングの必要がなく、マウスによる操作と最低限のパラメータ入力によって操作が完了する。アダプタ属性部品、マッピング部品の定義についても同様の手順で行う。

【００４２】図８にデータ形式部品選択処理フローを示す。８０１では、データ形式部品選択画面７００にて部品一覧を表示するために定義されたデータ形式部品の情報を読込む。読み込んだ情報を基に８０２にてデータ形式部品選択画面７００を表示する。ユーザの操作により８０３で処理を分岐する。

【００４３】処理８０３にてユーザが追加ボタン７０１をクリックした場合は、８０４にて新規データ形式部品を生成しユニークな部品ＩＤを自動的に割り当てる。８０５では、生成した部品の種別（データ形式）と部品ＩＤを連携情報部品一覧テーブル６００に追加する。その後に、８０７にてデータ形式部品属性定義画面７１０を表示し、ユーザに必要な部品属性を入力させる。ユーザが属性の入力を完了し該画面を閉じた場合は、入力した部品属性をデータ形式部品属性テーブル６２０に反映する。その後に、処理８０２に戻り、新規追加した部品も含めた部品一覧を再表示する。

【００４４】処理８０３にてユーザが編集ボタン７０２をクリックした場合は、一覧から指定された部品の属性を基に処理８０７にてデータ形式部品属性定義画面を表示する。新規作成と同様にユーザが属性編集を終了し該画面を閉じた場合は、修正した部品属性をデータ形式部品属性テーブル６２０に反映し処理８０２に戻る。

【００４５】処理８０３にてユーザが削除ボタン７０３をクリックした場合は、処理８０６にて一覧から指定された部品を連携情報部品一覧テーブル６００から削除する。そして、処理８０２に戻り、削除した部品を除いた部品一覧を再表示する。

【００４６】処理８０３にてユーザが所望の部品を選択し選択ボタン７０４をクリックした場合は、選択処理の結果としてユーザが選んだ部品のＩＤを返し処理を終了する。

【００４７】以上で説明した画面と処理により、連携情報部品を生成し、該部品をビジネスプロセスの構成要素に関連付けることが可能となる。但し、連携情報の定義については、ビジネスプロセスの構成要素と連携情報部品の関連付けだけでなく、部品間の関連付け定義も必要となる。例えば、アダプタ属性部品に入出力データとなるデータ形式部品を関連付けるといった定義がこれに該当する。そこで次に部品間の関連付けを定義する手順について説明する。

【００４８】図９に、アダプタ属性部品に業務プログラムの入出力データのデータ形式を示すデータ形式部品を関連付ける際に用いる画面を示す。

【００４９】アダプタ属性部品属性定義画面９００は、図７で説明したデータ形式部品属性定義画面７１０と同様に一つのアダプタ属性部品の属性を定義するための画面であり、部品名９０１、アダプタプログラムＩＤ９０３等をユーザに入力させる。ここで、対応する業務プログラムの出力データのデータ形式９０５を定義するために登録ボタン９０６をクリックすると、図７で説明したデータ形式部品選択画面７００が表示される。ユーザは、該画面に表示されているデータ形式部品の一覧から出力データの形式を示す部品を選択し選択ボタン７０４をクリックする。データ形式部品選択画面７００の表示が終了し９０５に選択したデータ形式部品のＩＤが表示される。入力データの形式に関しても同様の作業を行う。このような作業により部品間の関連付けが定義できる。

【００５０】図９で示したように、アダプタ属性部品の属性としてデータ形式部品を関連付けることも含めたアダプタ属性部品選択処理フローを図１０に示す。

【００５１】処理１００１～１００７では、図８で説明したデータ形式部品選択処理と同様にアダプタ属性部品選択画面の表示と、ユーザの入力に従った新規部品の生成、部品属性の編集、部品の削除を行い、最終的に選択された部品のＩＤを返して処理を終了する。但し、アダプタ属性部品選択処理では、処理１００７にてアダプタ属性部品属性定義画面９００を表示している際に、ユーザが業務プログラムの入出力データの形式を関連付ける処理が追加されている。ユーザが該画面にて登録ボタン９０６をクリックした場合は、図８で説明したデータ形式部品選択処理に遷移して目的のデータ形式部品を選択する（１００８）。ユーザが該画面にて編集ボタン９０７をクリックした場合は、図７に示したデータ形式部品属性定義画面７１０を表示してデータ形式部品の属性編集を行う（１００９）。

【００５２】このような処理フローにて、アダプタ属性部品の属性定義、及び、入出力データの形式を示すデータ形式部品の関連付けを行う。マッピング部品の属性定義、及び、変換元（先）データの形式を示すデータ形式部品の関連付けも同様の画面と処理フローにて行う。

【００５３】以上の説明で挙げた画面と処理にて、ビジ

ネスプロセスの制御と連携する業務プログラムを実行するために必要な、ビジネスプロセス定義情報１１６と、部品関連付け情報１２１と、連携情報部品の定義が完了する。この後、システム全体を稼動させるために、定義した情報を実行環境に分散配置する。分散配置の手順を、図１を用いて説明する。

【００５４】まず、ビジネスプロセス定義情報１１６をワークフロープログラム１３２が動作するワーク管理ノード１３０のディスク装置１３１に配置する（１３３）。

【００５５】次に、アダプタ属性部品１１７と、データ形式部品１１８と、マッピング部品１１９をアダプタプログラム１４２の動作する業務プログラム連携ノード１４０のディスク装置１４１に配置する（１４３）。

【００５６】最後に、部品関連付け情報１２１よりアダプタプログラム１４２が動作する際に必要なアダプタ運用情報１４４を生成し、業務プログラム連携ノード１４０のディスク装置１４１に配置する。該アダプタ運用情報１４４は、業務プログラム連携情報定義プログラム１１５によって自動生成される。

【００５７】上記アダプタ運用情報１４４の構造を、図１１に示す。該アダプタ運用情報１４４は、ビジネスプロセスから呼び出されるアダプタプログラム１４２が自身の実行時の方式や入出力データ変換に関する情報を入手するために用いる情報であり、アダプタ運用情報テーブル１１００によって構成される。該テーブルの各行は、図３で説明した業務ノード連携属性テーブル３４０の各行の定義内容を基に自動生成される。

【００５８】アダプタ運用情報テーブル１１００は、アダプタプログラム１４２を起動する際に指定する運用時ＩＤ１１０１と、起動した該アダプタプログラムの実行時属性を持つアダプタ属性部品のＩＤ１１０２と、対応する業務プログラムへの入力データに関するデータマッピング情報を持つマッピング部品のＩＤ１１０３と、同じく出力データのマッピング部品のＩＤ１１０４から構成される。アダプタ運用ＩＤ１１０１は、各行を生成する際に業務プログラム連携情報定義プログラム１１５によって自動生成され、残りの項目は業務ノード連携属性テーブル３４０の該当する行の値から生成される。

【００５９】必要な定義情報、及び、プログラムを配置した後にシステムを実行する。まず、アダプタプログラム１４２を起動する。この際、上記アダプタ運用情報テーブル１１００に格納したアダプタ運用ＩＤを引数として渡す。アダプタプログラム１４２は渡されたＩＤを基に運用情報を参照する。実行時の処理フローを図１２に示す。

【００６０】処理１２０１～１２０４は、ワーク管理ノード１３０のワークフロープログラム１３２がビジネスプロセスを制御する処理フローである。１２０１では、ビジネスプロセス定義情報１３３に定義された遷移を順番にたどり、全ての業務ノードに関する業務処理が終了した時点でビジネスプロセス全体の制御を終了する。１２０１で未実行の業務ノードがある場合は、処理１２０２に遷移し対象業務ノードの情報を取得する。１２０３では、図２で説明した業務ノードテーブルを参照し、取得した業務ノードに対応するアダプタプログラム１４２を呼び出す。１２０４にて目的の業務プログラムを実行したアダプタプログラムからの終了通知を検知して処理１２０１に戻る。

【００６１】処理１２１１～１２１９は、業務プログラム連携ノード１４０のアダプタプログラム１４２の処理フローである。処理１２１１は、アダプタ運用情報１４４より自身の実行に必要な情報を取得する。１２１２では、取得したアダプタ運用情報の入力マッピング部品ＩＤより対応する連携情報部品１４３（マッピング部品）を取得する。１２１３では、取得したマッピング部品より変換元（先）データの形式を示すデータ形式部品を取得する。１２１４では、ワーク管理ノード１３０の業務データＤＢ１３４より業務データを取得し、１２１３で取得したデータ形式部品と、１２１２で取得したマッピング部品を基に業務プログラムに渡すデータを生成する。１２１５では、実際の業務処理を行う業務プログラム１５１を起動し、１２１４で生成した入力データを渡す。業務プログラムの実行が終了した後、１２１６では１２１１で取得したアダプタ運用情報の出力マッピング部品ＩＤよりマッピング部品を取得する。１２１７では、取得したマッピング部品より変換元（先）データの形式を示すデータ形式部品を取得する。１２１８では、業務プログラムが出力したデータと１２１６、１２１７で取得したマッピング部品とデータ形式部品を基に業務データＤＢ１３４に格納する業務データを生成する。１２１９では、生成した業務データを業務データＤＢ１３４に格納し、アダプタプログラム１４２の実行を終了する。

【００６２】以上で説明した実施例により、個々の業務処理を行う業務プログラムをビジネスプロセスの遷移に伴い実行するための情報を生成し、該生成した情報を基にシステムを実行することが可能になる。

【００６３】本発明では、図１３に示すように、一つの連携情報部品を同一ビジネスプロセスの複数の業務ノードに関連付けることが可能である。また、図１４に示すように、部品間の関連付けにおいても連携情報部品の共有が可能である。図１３では、業務ノード１３０１と業務ノード１３０２で同じ業務プログラム１３０４を使用するため、「Ａ１」というアダプタ属性部品１３０３をどちらからも関連付けている例である。図１４では、アダプタ属性部品１４０１～１４０３がそれぞれデータ形式部品１４１１～１４１３を関連付けている例である。このように、連携情報部品を同一ビジネスプロセス内で共有することが可能なため、同様な定義作業の重複を回

避することができる。本発明では、次に示す実施例により、上記連携情報部品を異なる複数のビジネスプロセス間で共有することが可能となる。

【００６４】次に、本発明の第２の実施例（実施例２）について、説明する。図１５に連携情報部品を異なるビジネスプロセスで共有する場合のシステム構成を示す。図１にて説明したように、ビジネスプロセス制御、及び、業務プログラムの連携実行のために、ネットワーク１０１にワーク管理ノード１３０と、業務プログラム連携ノード１４０と、業務プログラムノード１５０を接続する。また、各ビジネスプロセスや連携情報を定義するための開発ノード１～開発ノードＮを接続する（１５０１、１５０２）。各開発ノードは開発ノード１と同様のプログラムと定義情報を持つ。更に、ビジネスプロセスとは独立した連携情報部品を管理するためのパブリック部品管理ノード１５１０から構成される。

【００６５】パブリック部品管理ノード１５１０は、異なるビジネスプロセス間で共有するパブリック部品の情報を格納するディスク装置１５１１と、パブリック部品情報の受け渡しを行うパブリック部品管理プログラム１５１２から構成され、ディスク装置１５１１には、実施例１で説明したアダプタ属性部品１５１３、データ形式部品１５１４、マッピング部品１５１５を格納するパブリック部品庫１５１６と、該パブリック部品庫に格納された連携情報部品の一覧情報を持つパブリック部品一覧情報１５１７を格納する。

【００６６】ここでは、連携情報部品を異なる複数のビジネスプロセス間で共有するための手順を説明する。

【００６７】まず、開発ノード１において、実施例１で説明した手順により開発ノード１で管理するビジネスプロセスに関連する連携情報部品を定義する。ここで定義された部品は、プライベート部品庫１２０に格納されたプライベート部品である。

【００６８】例えば、データ形式部品「Ｆ１」をビジネスプロセス間で共有する場合は、該データ形式部品と同等の定義内容を持つ複製部品「ＥＦ１」を生成する。そして、生成した複製部品を、どの開発ノードからも参照できるパブリック部品として、パブリック部品管理ノード１５１０のパブリック部品庫１５１１に格納する。該プライベート部品からパブリック部品を生成する処理をエクスポート処理と呼び、複製部品の生成やパブリック部品庫への格納はユーザによる画面上のアイコン操作に基づき自動的に行われる。続いて、開発ノードＮにて、上記パブリック部品庫１５１６に格納したデータ形式部品「ＥＦ１」を流用する場合は、該パブリック部品の複製部品「Ｆ２」を生成し、開発ノードＮのプライベート部品庫に生成した部品を格納する。該パブリック部品からプライベート部品を生成する処理をインポート処理と呼び、エクスポート処理と同様に複製部品の生成やプライベート部品庫への格納はユーザの指示に基づき自動的に行われる。このようにプライベート部品のエクスポート処理、及び、パブリック部品のインポート処理といった二つの手段を提供することにより、定義した連携情報部品を異なる複数のビジネスプロセスで共有することが可能となる。

【００６９】図１６に、パブリック部品を操作する際の連携情報部品選択画面１６００を示す。本実施例では、実施例１で説明した連携情報部品の選択に図１６に示すような画面を用いる。画面上部には、実施例１と同様にプライベート部品の一覧を示す（１６０１）。一方、画面下部にはパブリック部品庫内に存在するパブリック部品の一覧を表示する（１６０２）。例えば、ユーザがパブリック部品「ＥＦ４」をインポートする場合は、図に示すように部品「ＥＦ４」を示すアイコンをドラッグして、プライベート部品一覧（１６０１）にドロップする。ユーザがドラッグした「ＥＦ４」と同様の属性を持つプライベート部品が生成され、プライベート部品一覧に追加表示される。エクスポートの場合は、インポートと逆に目的のプライベート部品のアイコンをドラッグし、パブリック部品の一覧（１６０２）にドロップする。最後にプライベート部品の中から所望の部品を選択し選択ボタン１６０３をクリックする。本画面の表示、及び、処理は、開発ノードの業務プログラム連携情報定義プログラムが行う。

【００７０】図１７に連携情報部品のエクスポート・インポートを行う場合の部品選択処理フローを示す。処理１７０１では、実施例１に示した連携情報部品一覧テーブル６００と上記パブリック部品一覧情報１５１７よりプライベート部品とパブリック部品の一覧情報を取得する。パブリック部品一覧情報１５１７は、連携情報部品一覧テーブル６００と同様のテーブルから構成される。処理１７０２では、図１６に示した画面を表示する。処理１７０３にてユーザが追加ボタン１６０４、編集ボタン１６０５、削除ボタン１６０６をクリックした場合は、実施例１で説明した処理を実行し、処理１７０２に戻って部品一覧を再表示する。ここで選択ボタン１６０３を選択した場合は処理１７０７に遷移する。プライベート部品、あるいは、パブリック部品の一覧からアイコンをドラッグ＆ドロップした場合は処理１７０４に遷移する。処理１７０４では、ドラッグされた部品の種別を判定し、ドラッグされた部品がプライベート部品の場合は処理１７０５に、パブリック部品の場合は処理１７０６に遷移する。処理１７０５では、ドラッグされたプライベート部品を基にパブリック部品を生成するエクスポート処理を実行する。処理１７０６では、ドラッグされたパブリック部品を基にプライベート部品を生成するインポート処理を実行する。処理１７０５、１７０６を実行後、処理１７０２に戻り部品一覧を再表示する。処理１７０３にて選択ボタン１６０３をユーザが選択し処理１７０７に遷移した場合は、選択された部品の種別を判

定する。選択された部品がパブリック部品の場合は、その部品選択を無効とし処理１７０２に戻る。選択された部品がプライベート部品の場合は、選択された部品のＩＤを返し部品選択処理全体を終了する。

【００７１】エクスポート処理では、処理１７１１にてドラッグされたプライベート部品の情報を取得する。処理１７１２では、取得したプライベート部品の情報を基に同様の属性を持つ新規部品を生成する。処理１７１３では、生成した複製部品をパブリック部品管理プログラム１５１２に依頼してパブリック部品庫１５１６に格納する。処理１７１４では、生成した部品の一覧情報をパブリック部品管理プログラム１５１２に依頼してパブリック部品一覧情報１５１７に追加する。

【００７２】インポート処理では、処理１７２１にてドラッグされたパブリック部品の情報をパブリック部品管理プログラム１５１２に問合せて取得する。処理１７２２では、取得したパブリック部品の情報を基に同様の属性を持つ新規部品を生成する。処理１７２３では、生成した複製部品をプライベート部品庫１２０に格納する。処理１７２４では、生成した部品の一覧情報を連携情報部品一覧テーブル６００に追加する。

【００７３】以上で説明した実施例により、開発ノードで定義した連携情報部品をパブリック部品庫にエクスポートし、該パブリック部品を異なる開発ノードでインポートすることによって、異なる複数のビジネスプロセスにおいて部品の共有が可能になる。

【００７４】次に、本発明の第３の実施例（実施例３）について説明する。実施例１の図５の説明において、ユーザがビジネスプロセス上の構成要素に対する不当な連携情報部品の関連付けを拒否する手段について述べた。同様に本実施例で挙げる処理により、ビジネスプロセス制御、及び、業務プログラムの連携実行時における障害発生を回避することが可能となる。

【００７５】実施例１の図１３、図１４で示したように、本発明の連携情報部品は複数のビジネスプロセス要素、あるいは、複数の他の部品から関連付ける（共有する）ことが可能である。例えば、図１３に示した例では、業務ノード１３０１と業務ノード１３０２にアダプタ属性部品１３０３を関連付けている。しかしながら、もしこのような状況で業務ノード１３０１の属性として再度部品の属性を修正してしまった場合、その修正内容が業務ノード１３０２にも影響し、ユーザがこれを意図していない場合は実行時に障害を起こす可能性がある。そこで、このように一つの連携情報部品を複数のビジネスプロセス要素、あるいは、複数の他の部品から共有している場合は、該部品の属性編集、及び、該部品削除を拒否する処理を提供することによって、上記障害発生を回避することが可能となる。

【００７６】図１８に連携情報部品の選択処理において共有関係にある部品の削除・編集を拒否する処理フローを示す。ここで挙げる処理フローは、アダプタ属性部品、データ形式部品、マッピング部品で共通である。

【００７７】処理１８０１では、画面に部品一覧を表示するために連携情報部品一覧テーブル６００より部品の一覧情報を取得する。処理１８０２では、取得した一覧情報より部品選択画面を表示する。処理１８０３にてユーザが選択ボタンをクリックした場合は、選択処理全体を終了する。処理１８０３にてユーザが追加ボタンをクリックした場合は、処理１８０４に遷移して新規部品を生成する。処理１８０５では、新規生成した部品の情報を連携情報部品一覧テーブル６００に追加し、処理１８０６にて追加した部品の詳細属性を定義した後に処理１８０２に戻る。処理１８０３にてユーザが編集ボタンをクリックした場合は、処理１８０７に遷移して選択されている部品が共有状態にあるか判定する。該部品が共有状態にない場合は、処理１８０６に遷移して部品の詳細属性を編集し処理１８０２に戻る。該部品が共有状態にある場合は、処理１８０８にてユーザに共有状態であることを提示し、処理１８０９に遷移する。処理１８０９では、該部品の詳細属性を表示するが、ユーザからの属性編集は受け付けずに属性の閲覧のみを行い、閲覧が終了後に処理１８０２に戻る。処理１８０３にてユーザが削除ボタンをクリックした場合は、処理１８１０に遷移して選択されている部品が共有状態にあるか判定する。該部品が共有状態にない場合は、処理１８１２に遷移して指定部品を連携情報部品一覧テーブル６００より削除して処理１８０２に戻る。該部品が共有状態にある場合は、処理１８１１にてユーザに共有状態であることを提示し、部品の削除は行わず処理１８０２に戻る。

【００７８】図１９に、図１８の説明で挙げた部品の被関連付け探索処理のフローを示す。処理１９０１～１９０５にて、対象部品とビジネスプロセス構成要素との関連付けを探索し（１９００）、処理１９１１～１９１８にて対象部品と他の部品の関連付けを探索する（１９１０）。まず、対象部品とビジネスプロセス構成要素との関連付けを探索する処理について説明する。処理１９０１では、対象部品の種別を判定する。対象部品がデータ形式部品の場合は、処理１９０２に遷移して、業務データ部品テーブル３３０より対象部品が業務データに関連付けられていないか判定する。該部品が業務データに関連付けられている場合は、１９２１に遷移して「関連付けあり」として処理を終了する。該部品が業務データに関連付けられていない場合は、処理１９１１に遷移する。対象部品がアダプタ属性部品、あるいは、マッピング部品の場合は、処理１９０３に遷移して業務ノード連携属性テーブル３４０より１行分の情報を取得する。処理１９０４では、処理１９０３にて既にテーブル上の全ての行に対して処理を行った場合は処理１９１１に遷移し、未処理の行がある場合は処理１９０５に遷移する。処理１９０５では、対象部品が処理１９０３にて取得し

た業務ノードに関連付けられているか判定する。関連付けられていない場合は、処理１９０３に戻り、関連付けられていた場合は、１９２１に遷移して「関係づけあり」として処理を終了する。

【００７９】次に、対象部品と他の部品との関連付けを探索する処理について説明する。処理１９１１では、対象部品の種別を判定する。対象部品がデータ形式部品の場合は、処理１９１２に遷移する。処理１９１２では、連携情報一覧テーブル６００よりアダプタ属性部品、あるいは、マッピング部品の行を参照し該部品の属性テーブル情報を取得する。処理１９１３では、処理１９１２にて既にテーブル上の全ての行に対して処理を行った場合は、１９２２に遷移して「関連付けなし」として処理を終了し、未処理の行が残っている場合は処理１９１４に遷移する。処理１９１４では、対象部品が取得した部品のアダプタ入出力データ、あるいは、マッピング変換元（先）データとして関連付けられている場合は、１９２１に遷移して「関連付けあり」として処理を終了し、関連付けがない場合は、処理１９１２に戻る。処理１９１１で対象部品がアダプタ属性部品の場合は、処理１９１５に遷移する。処理１９１５では、対象部品の入出力データに関連付けたデータ形式部品のＩＤを取得する。処理１９１６では、連携情報一覧テーブル６００のマッピング部品の行より該マッピング部品の属性テーブルを参照して詳細属性を取得する。処理１９１７では、処理１９１６で既に全てのマッピング部品について処理した場合は、１９２２に遷移して「関連付けなし」として処理を終了し、未処理のマッピング部品がある場合は、処理１９１８に遷移する。処理１９１８では、アダプタ部品の入出力データに関連付けたデータ形式部品が該マッピング部品の変換元（先）データとして関連付けられていないか判定する。マッピング部品の変換元（先）データとして関連付けられていた場合は、１９２１に遷移して「関連付けあり」として処理を終了し、関連付けられていいない場合は、処理１９１６に戻る。処理１９１１で対象部品がマッピング部品の場合は、１９２２に遷移して「関連付けなし」として処理を終了する。

【００８０】以上本実施例で説明したように、共有状態にある連携情報部品の削除・属性編集を自動的に拒否する手段を提供することにより、ビジネスプロセス制御、及び、業務プログラムの連携実行時における障害発生を未然に防ぐことが可能となる。

【００８１】次に、本発明の第４の実施例（実施例４）について、説明する。これまでに述べた実施例により、業務プログラムの連携情報を生成し、ビジネスプロセスの制御、及び、業務プログラムの連携実行が可能である。しかしながら、ユーザが必要な連携情報部品の定義や該部品の関連付けを忘れた場合は、実行時に障害が発生する可能性がある。このような、連携情報定義の不足による実行時障害発生を回避するために、本実施例ではユーザの定義した連携情報を検証する処理について説明する。

【００８２】実行時障害が発生しないために、次に挙げる６項目が的確に定義されていることが必要となる。アダプタ属性部品が業務ノードに関連付けられている場合に、（１）該部品にアダプタプログラムＩＤが設定されていること、（２）該部品に入出力データが在る場合は業務データにデータ形式部品が関連付けられていること、（３）該部品の入出力データに対応するマッピングが定義されていること、（４）（３）のマッピング部品の入出力データＩＤと変換データＩＤが同一であること、（５）（３）のマッピング部品の業務データ側変換データＩＤが（２）のデータ形式部品ＩＤと同一であること、（６）（３）のマッピング部品にデータ項目の対応が定義されていること、が必要である。

【００８３】図２０、及び、図２１に上記条件を検証する連携情報定義の検証処理フローを示す。本処理は、ユーザの指示により業務プログラム連携情報定義プログラム１１５によって実行される処理であり、ユーザが連携情報の定義を終了し、定義情報を実行環境に分散配置する直前に実行すればよい。

【００８４】変数ＰＤ＿ＦＬＡＧ（２０２１）は、業務データに対応するデータ形式部品が関連付けられていることを示すフラグである。変数ＰＤ＿ＩＤ（２０２２）は、業務データの形式を表すデータ形式部品のＩＤである。変数ＥＲＲ＿ＭＳＧ（２０２３）は、検証処理中に定義エラーに関するメッセージ文字列を蓄積する変数であり、本処理の最後にユーザに検証結果を提示するために用いる。

【００８５】処理２００１では、変数ＰＤ＿ＦＬＡＧを初期化する。処理２００２では、業務データ部品テーブル３３０を参照し、変数ＰＤ＿ＩＤに業務データの形式を表すデータ形式部品のＩＤを代入する。該テーブルにＩＤが未定義の場合は、ＮＵＬＬを設定する。処理２００３では、業務ノード連携属性テーブル３４０を参照し、業務ノード一つに関連する情報を取得する。処理２００３で既に全ての業務ノードに関する検証が終了している場合は、２００４より処理２００５に遷移し、変数ＥＲＲ＿ＭＳＧに蓄積したエラーメッセージをユーザに提示して処理を終了する。処理２００３で未検証の業務ノードがある場合は、処理２００６に遷移する。処理２００６では、処理２００３で取得した情報にあるアダプタ属性部品ＩＤより該部品情報を取得する。処理２００７では、取得したアダプタ属性部品にアダプタプログラムＩＤが設定されているか検証し、設定されていない場合は処理２００８に遷移してエラーメッセージを変数ＥＲＲ＿ＭＳＧに蓄積する。処理２００９では、取得したアダプタ属性部品に入出力データが定義されていない場合、処理２００３に戻る。アダプタ属性部品に入出力データが定義している場合は、処理２０１０に遷移する。

処理２０１０では、変数ＰＤ＿ＦＬＡＧがＯＦＦのままであり、かつ、変数ＰＤ＿ＩＤにデータ形式部品のＩＤが設定されずにＮＵＬＬとなっている場合は、処理２０１１に遷移して業務データに対応するデータ形式部品の未定義メッセージを変数ＥＲＲ＿ＭＳＧに蓄積する。処理２０１２では、変数ＰＤ＿ＦＬＡＧをＯＮに設定し処理２１０１に遷移する。

【００８６】処理２１０１では、処理２００３で取得した情報にある入出力マッピング部品ＩＤより該部品情報を取得する。処理２１０２では、処理２１０１で取得したマッピング部品を基に、処理２００６で取得したアダプタ属性部品の入出力データに対応するマッピング部品が業務ノードに関連付けられているか検証する。適当なマッピング部品が関連付けられていない場合は、処理２１０３にて変数ＥＲＲ＿ＭＳＧにエラーメッセージを蓄積し、処理２００３に戻る。処理２１０４では、処理２００６で取得したアダプタ属性部品の入出力データＩＤと、処理２１０１で取得したマッピング部品の対応する変換データＩＤが同一であることを検証する。ＩＤが異なる場合は、処理２１０５にて変数ＥＲＲ＿ＭＳＧにエラーメッセージを蓄積し、処理２００３に戻る。処理２１０６では、処理２１０１で取得したマッピング部品の業務データ側の変換データＩＤが変数ＰＤ＿ＩＤと同一であることを検証する。ＩＤが異なる場合は、処理２１０７にて変数ＥＲＲ＿ＭＳＧにエラーメッセージを蓄積し、処理２００３に戻る。処理２１０８では、処理２１０１で取得したマッピング部品にデータ項目マッピングテーブル６３６が定義されていることを検証する。データ項目の対応関係が定義されていない場合は、処理２１０９にて変数ＥＲＲ＿ＭＳＧにエラーメッセージを蓄積し処理２００３に戻り、該テーブルが定義されている場合は、そのまま処理２００３に遷移する。

【００８７】本実施例の処理を提供することにより、ユーザの連携情報定義の不足による実行時障害の発生を自動的に未然に防ぐことが可能となる。

【００８８】以上で説明したように、本発明の各実施例によれば、次に挙げる効果が得られる。

（１）ビジネスプロセスの業務ノードに連携する業務プログラム連携情報を分類し細かな単位で部品化することで、連携情報の多重定義を回避しシステムの早期構築を可能にする。また、誤った定義の削減にも寄与する。

（２）業務プログラムが受け渡すデータの形式をビジネスプロセスとは独立に定義・管理するため、外部システムで定義した既存データ形式を取り込むことが容易である。

（３）業務プログラム連携情報とビジネスプロセスの業務ノードとの関連付けは、ビジネスプロセス定義画面における対象の属性として選択するユーザインタフェースを提供することで、ユーザがその関連性を明確に把握することができる。また、ユーザは業務プログラム連携定義に関して特別なプログラミングを必要としない。

（４）定義した連携情報部品をビジネスプロセスとは独立したパブリック部品庫に格納・取り出しできることで、同一ビジネスプロセス内だけに限らず、異なる複数のビジネスプロセス間において部品の流用が可能である。

（５）業務プログラム連携情報をビジネスプロセス定義とは独立させることで、目的の業務プログラムを連携するための情報（連携情報部品やアダプタプログラム）を、ビジネスプロセスを定義するよりも前に予め定義・開発することが可能で、システム構築者は提供された部品の関連付けだけを行えばよい。

【００８９】

【発明の効果】本発明によれば、業務プログラムを容易に構築することが可能になる。

【図面の簡単な説明】

【図１】第１の発明のシステム構成である。

【図２】１１４にて表示するビジネスプロセスを定義する画面と生成するデータの例である。

【図３】１１５が生成するプログラム連携データとデータ間の関連を示した例である。

【図４】１１５にて表示するプログラム連携データの関連付けを行う画面の例である。

【図５】１１５のプログラム連携データ関連付け処理の流れ図である。

【図６】１１５が生成するプログラム連携データの例である。

【図７】１１５にて表示するプログラム連携データ（データ形式）の詳細定義を行う画面の例である。

【図８】１１５のプログラム連携データ（データ形式）選択処理の流れ図である。

【図９】１１５にて表示するプログラム連携データ（アダプタ属性）の詳細定義を行う画面の例である。

【図１０】１１５のプログラム連携データ（アダプタ属性）選択処理の流れ図である。

【図１１】１１５が生成するアダプタプログラム運用データの例である。

【図１２】１３２と１４２のビジネスプロセス制御及びプログラム連携処理の流れ図である。

【図１３】１１５が生成するプログラム連携データの関連付けを示した例である。

【図１４】１１５が生成するプログラム連携データの関連付けを示した例である。

【図１５】第４の発明のシステム構成である。

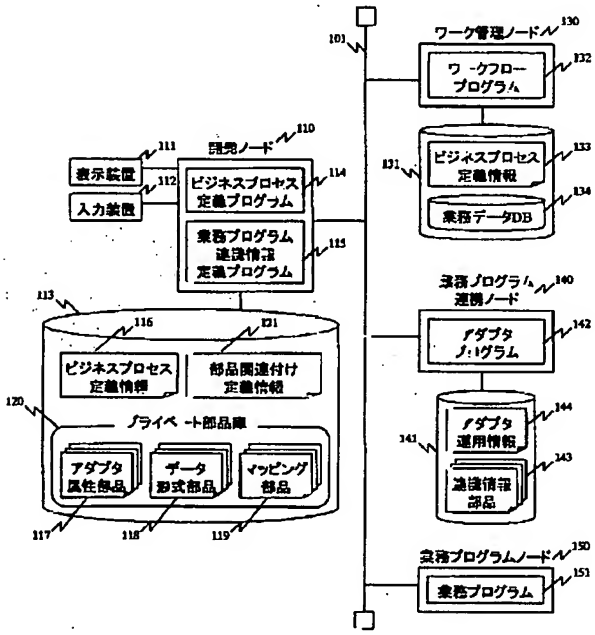
【図１６】１１５が表示するプログラム連携データの流用を行う画面の例である。

【図１７】１１５がプログラム連携データの流用を行う処理の流れ図である。

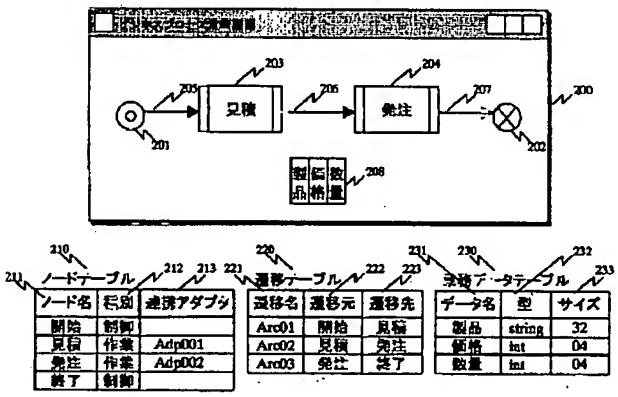
【図１８】１１５の不当なユーザの操作を回避する処理の流れ図である。

【図19】115がプログラム連携データの共有参照関係を探索する処理の流れ図である。
【図20】115がユーザの定義した情報の妥当性を検証する処理の流れ図である。
【図21】115がユーザの定義した情報の妥当性を検証する処理の流れ図である。
【符号の説明】
101　ネットワーク、111　表示装置、112　入力装置、114　ビジネスプロセス定義プログラム、115　業務プログラム連携情報定義プログラム、116、133　ビジネスプロセス定義情報、121　部品関連付け情報、117～119、143　連携情報部品、132　ワークフロープログラム、134　業務データDB、142　アダプタプログラム、144　アダプタ運用情報、151　業務プログラム、200　ビジネスプロセス定義画面、210、220、230　ビジネスプロセス定義情報を構成するテーブル、330、340　部品関連付け情報を構成するテーブル、400　連携情報部品選択画面、600、610、620、630、636　連携情報部品を構成するテーブル、710　データ形式部品属性定義画面、900　アダプタ属性部品属性定義画面、1100　アダプタ運用情報を構成するテーブル、1512　パブリック部品管理プログラム、1517　パブリック部品一覧情報、1600　パブリック部品の一覧表示を含めた連携情報部品選択画面

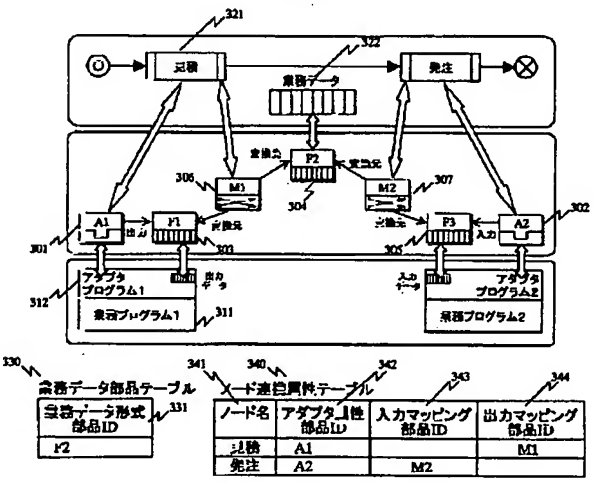
【図1】



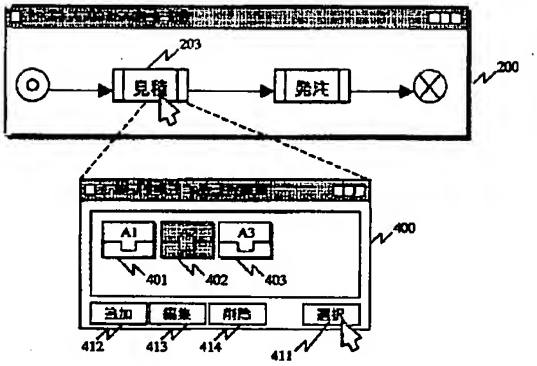
【図2】



【図3】

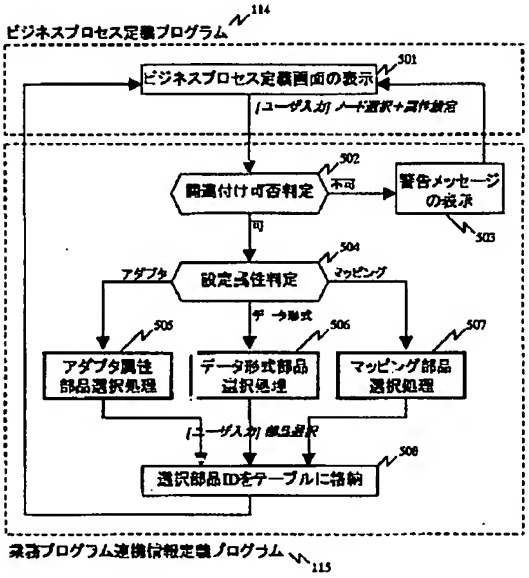


【図4】



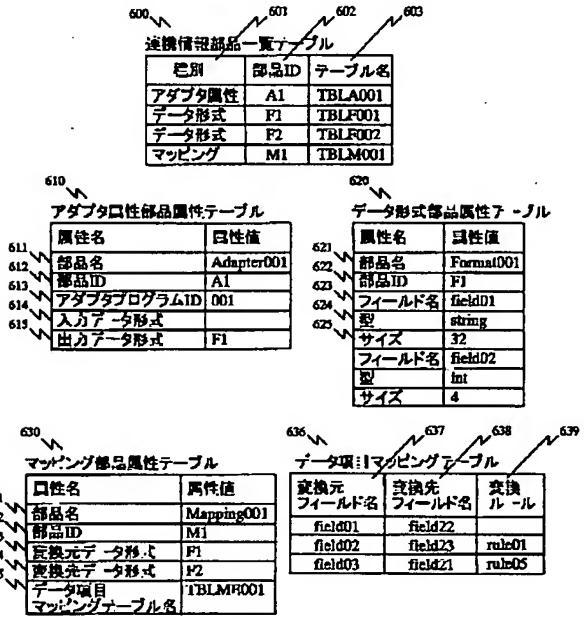
【図5】

【図5】



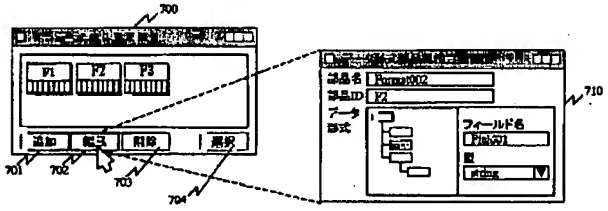
【図6】

【図6】



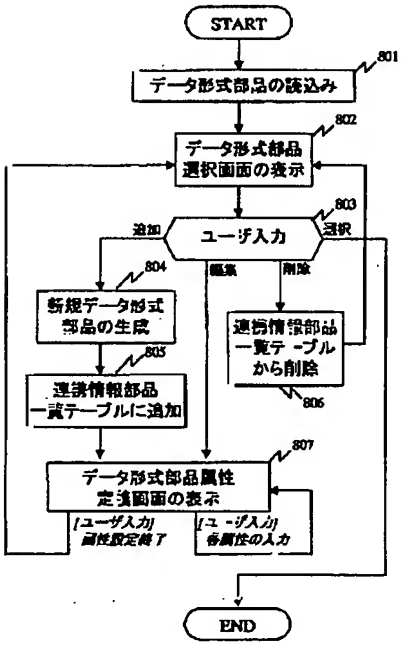
【図7】

【図7】



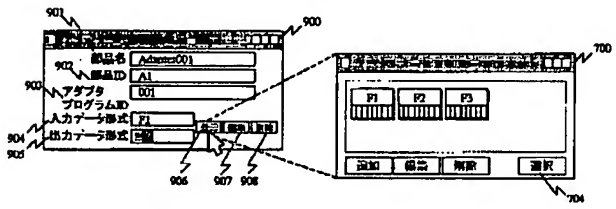
【図8】

【図8】

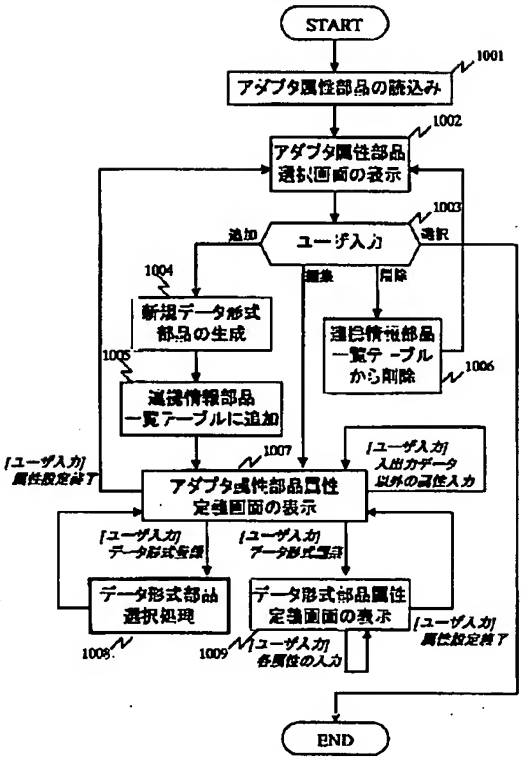


【図9】

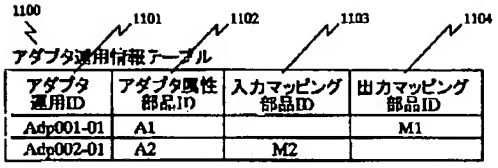
【図9】



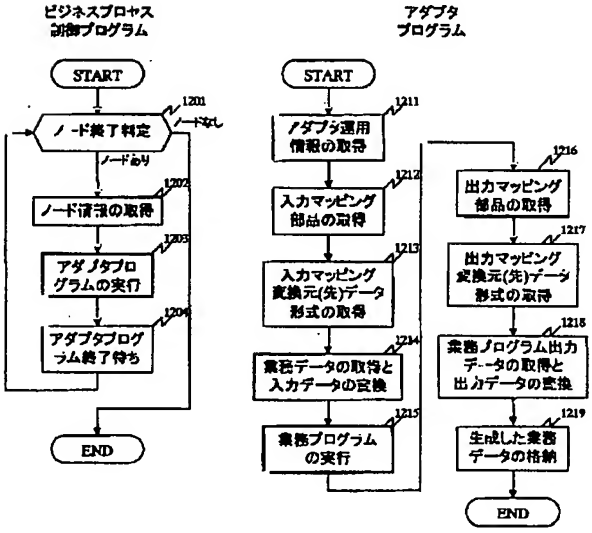
【図10】



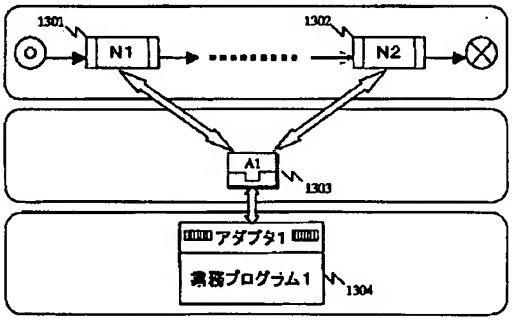
【図11】



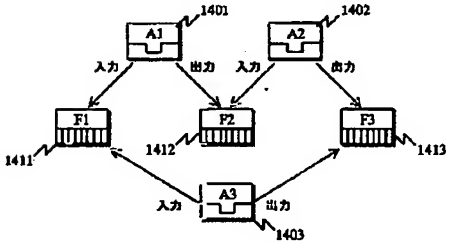
【図12】



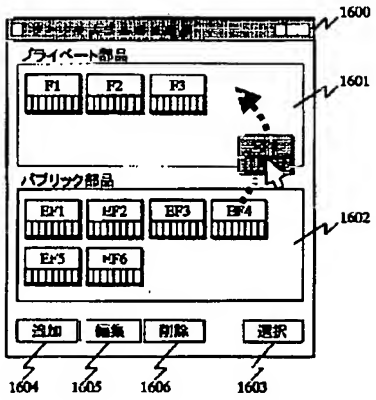
【図13】



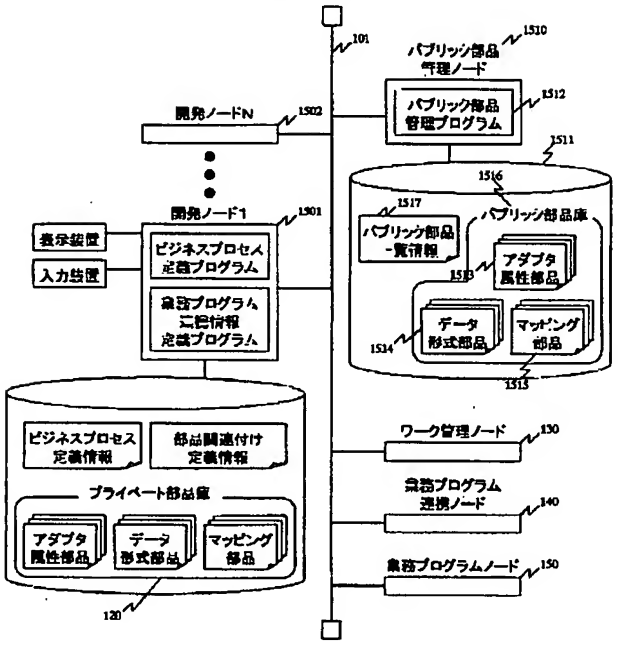
【図14】



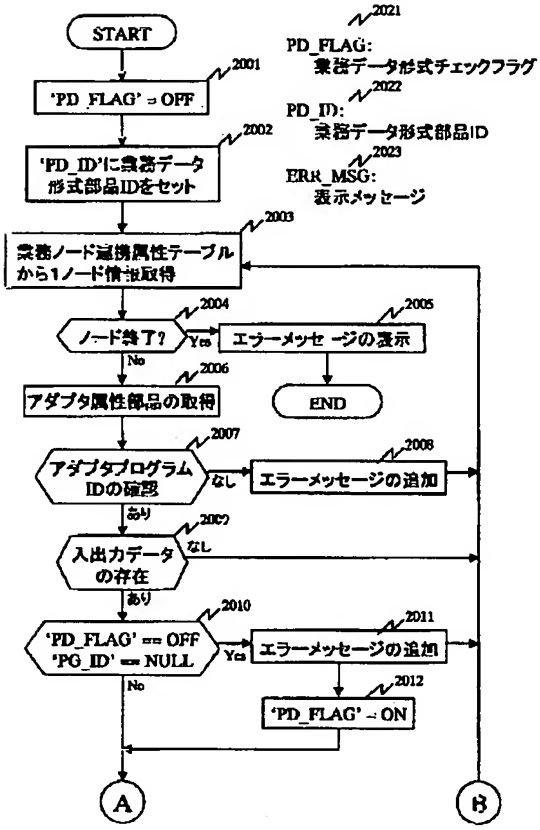
【図16】



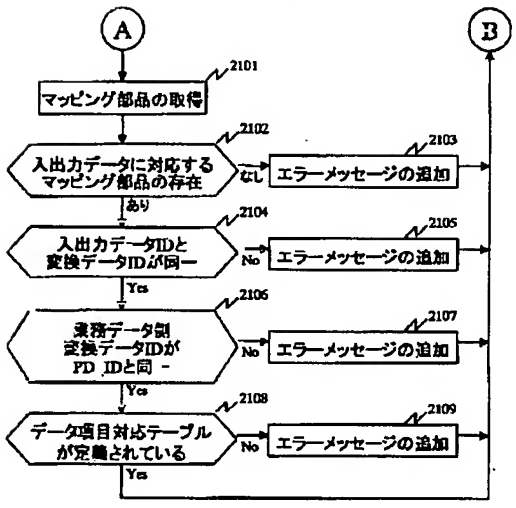
【図15】


101
パブリック部品管理ノード
1510
パブリック部品管理プログラム
1512
開発ノードN
1502
1511
1516
開発ノード1
1501
1517
パブリック部品一覧情報
パブリック部品庫
1513
アダプタ属性部品
表示装置
入力装置
ビジネスプロセス定義プログラム
業務プログラム仕様情報定義プログラム
1514
データ形式部品
マッピング部品
1515
ビジネスプロセス定義情報
部品関連付け定義情報
プライベート部品庫
アダプタ属性部品
データ形式部品
マッピング部品
120
ワーク管理ノード
130
業務プログラム連携ノード
140
業務プログラムノード
150

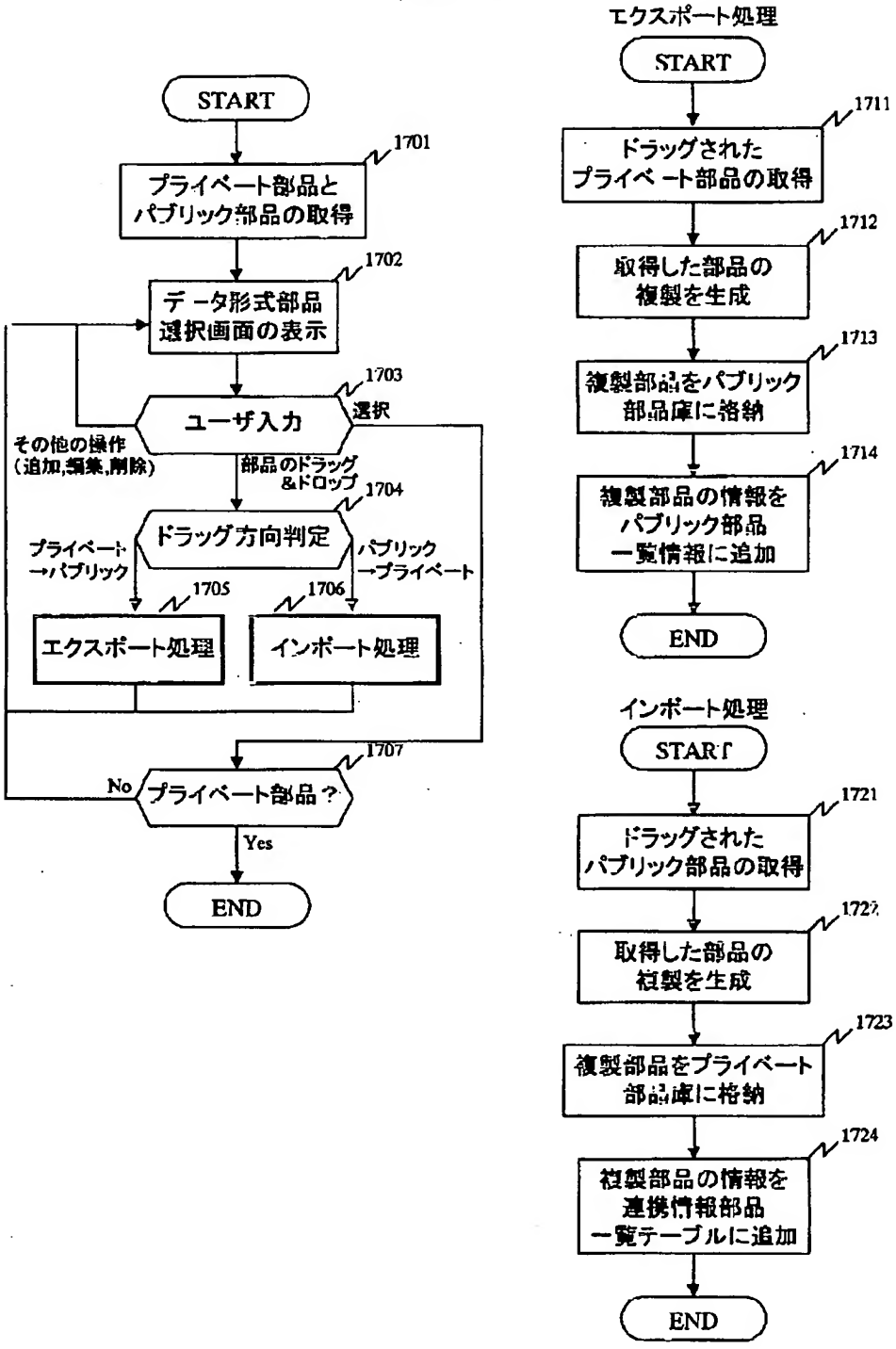
【図20】


START
2021
PD_FLAG:
業務データ形式チェックフラグ
2001
'PD_FLAG' = OFF
2022
PD_ID:
業務データ形式部品ID
2002
'PD_ID'に業務データ形式部品IDをセット
2023
ERR_MSG:
表示メッセージ
2003
業務ノード連携属性テーブルから1ノード情報取得
2004
ノード終了?
Yes
2005
エラーメッセージの表示
No
2006
END
アダプタ属性部品の取得
2007
アダプタプログラムIDの確認
なし
2008
エラーメッセージの追加
あり
2009
入出力データの存在
なし
あり
2010
'PD_FLAG' == OFF
'PG_ID' == NULL
Yes
2011
エラーメッセージの追加
No
2012
'PD_FLAG' = ON
A
B

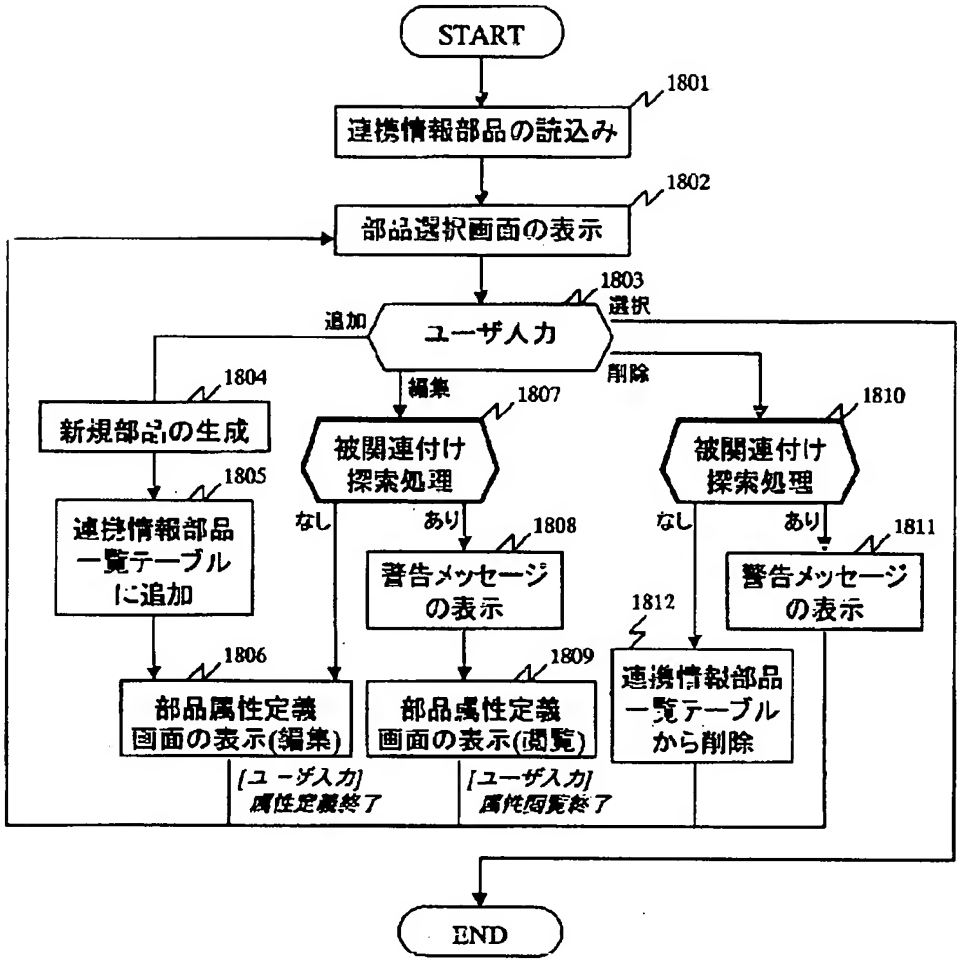
【図21】


A
B
2101
マッピング部品の取得
2102
入出力データに対応するマッピング部品の存在
なし
2103
エラーメッセージの追加
あり
2104
入出力データIDと変換データIDが同一
No
2105
エラーメッセージの追加
Yes
2106
業務データ側変換データIDがPD_IDと同一
No
2107
エラーメッセージの追加
Yes
2108
データ項目対応テーブルが定義されている
No
2109
エラーメッセージの追加
Yes

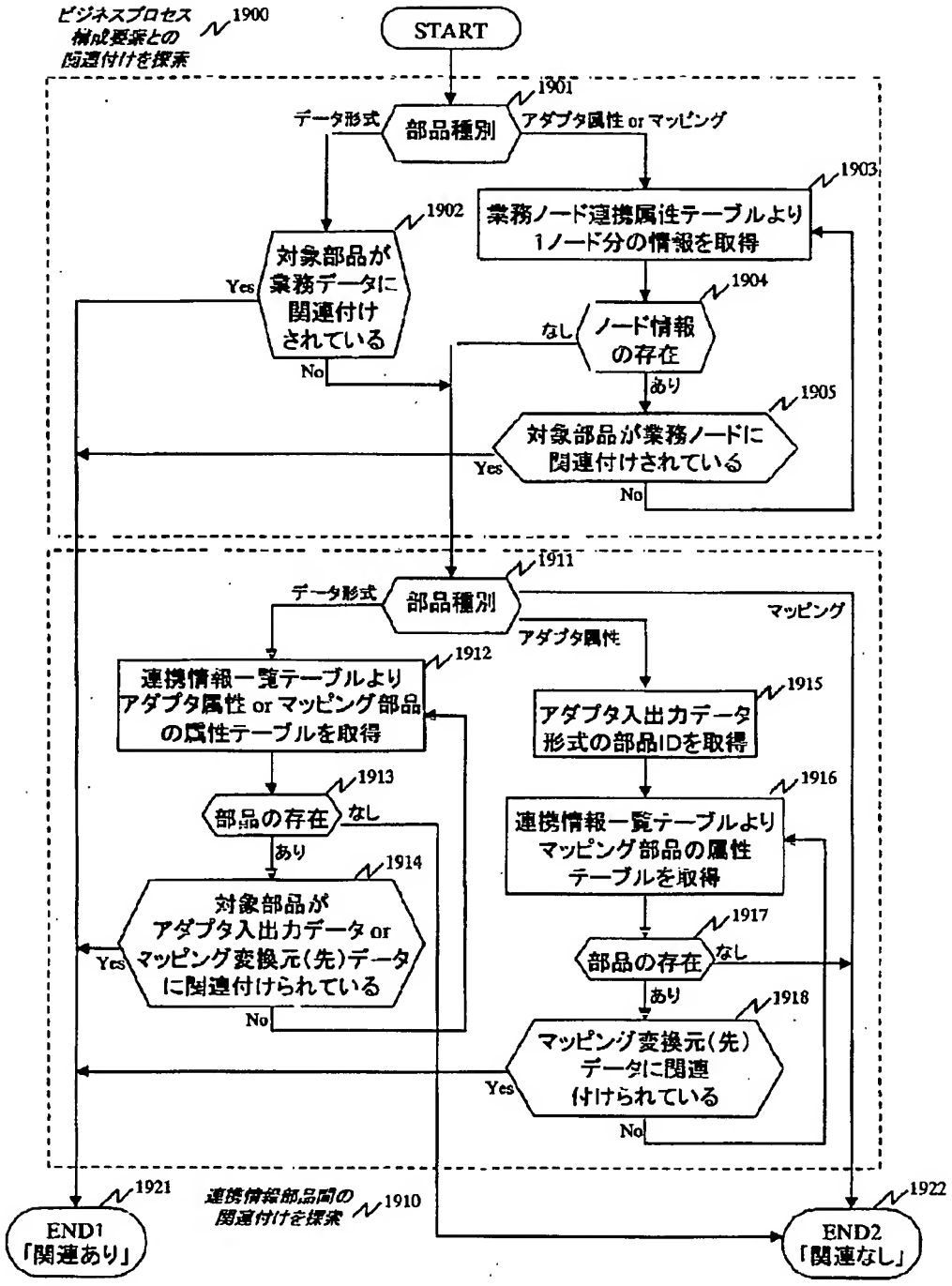
【図17】


START
1701
プライベート部品と
パブリック部品の取得
1702
データ形式部品
選択画面の表示
1703
ユーザ入力
選択
その他の操作
(追加,編集,削除)
部品のドラッグ
&ドロップ
1704
ドラッグ方向判定
プライベート
→パブリック
パブリック
→プライベート
1705
1706
エクスポート処理
インポート処理
1707
No
プライベート部品？
Yes
END
エクスポート処理
START
1711
ドラッグされた
プライベート部品の取得
1712
取得した部品の
複製を生成
1713
複製部品をパブリック
部品庫に格納
1714
複製部品の情報を
パブリック部品
一覧情報に追加
END
インポート処理
START
1721
ドラッグされた
パブリック部品の取得
1722
取得した部品の
複製を生成
1723
複製部品をプライベート
部品庫に格納
1724
複製部品の情報を
連携情報部品
一覧テーブルに追加
END

【図18】


START
1801
連携情報部品の読込み
1802
部品選択画面の表示
1803
選択
追加
ユーザ入力
編集
削除
1804
新規部品の生成
1807
被関連付け探索処理
1810
被関連付け探索処理
1805
連携情報部品一覧テーブルに追加
なし
あり
1808
警告メッセージの表示
なし
あり
1811
警告メッセージの表示
1812
1806
部品属性定義画面の表示(編集)
1809
部品属性定義画面の表示(閲覧)
連携情報部品一覧テーブルから削除
[ユーザ入力]
属性定義終了
[ユーザ入力]
属性閲覧終了
END

【図19】



フロントページの続き

(72)発明者　田中　哲雄
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

Fターム(参考)　5B076 DB04 DC09 DC10 DD05 DD07 DD08 DF06 DF08